# A Load Balancing Method for Range-Queryable Structured Overlays

Youki Shiraishi[1] and Michiko Harayama[2]

[1] Graduate School of Information Science, Nara Institute of Science and Technology, 8916–5 Takayama, Ikoma, Nara 630–0192, Japan,
shiraishi@computer.org
[2] Faculty of Engineering, Gifu University, 1–1 Yanagido, Gifu City 501-1193, Gifu 501–1193, Japan

**Abstract.** When implementing a key-value store using a range-queryable structured overlay, it is necessary to correspond one-to-one with a network node and a data object, and difficulties arise when the number of objects handled for a node is large. In this paper, we propose a method of assigning multiple objects to each node of range-queryable structured overlays, and balancing the load of each node.

**Key words:** pervasive computing, peer-to-peer computing, distributed hash tables, load balancing

## 1 Introduction

The overlay network is a logical network constructed on top of the existing network such as IP network. In a wide area distributed system where many nodes cooperate, it is indispensable to build an overlay in an autonomous and distributed manner in order to improve fault tolerance and scalability. Distributed hash table (DHT) is one of the applications of structured overlay represented by Chord [1]. DHT equally assigns keys to each node using consistent hashing [2], and constructs an overlay capable of searching for keys. In the DHT search space, since the network topology based on the hash value has destroyed the key order relationship, complicated query such as range query is impossible. On the other hand, the structured overlay represented by the Skip Graph [3] builds the topology that keeps the order relation of the keys based on the Skip List [4], enabling the range query. Since the structured overlay based on the Skip List is assuming that the node and the key correspond one to one, load balancing cannot be taken into account.

In this paper, we propose a method to balance the load by equally assigning multiple objects to each node in range-queryable structured overlays. In the proposed method, an overlay is designed by dividing it into a layer that associates a node with an object and a layer that actually performs queries. First, some of objects are equally assigned to each node based on consistent hashing. Then, an overlay which supports range search is constructed based on the assigned key.

## 2 Proposed Method

The proposed method makes correspondence between nodes and objects based on consistent hashing. Nodes and objects have $m$ bits identifiers. The node ID and the object ID are the hash value of the IP address and the hash value of the key, respectively. Nodes and objects are ordered on the ID-based ring-like identifier space $\mathcal{I} = \mathbb{Z}/2^m\mathbb{Z}$. ID $x$ is assigned to the first node found on $\mathcal{I}$, proceeding in the positive direction from $x$. This node is called the successor of ID $x$ and is written as $\mathrm{succ}^N(x)$, where $N$ is the set of nodes on the overlay. Also, the node found first in the counterclockwise direction from ID $x$ is called the predecessor of ID $x$ and written $\mathrm{pred}^N(x)$. Each node $n$ holds path information to $\mathrm{succ}^N(x)$, $\mathrm{pred}^N(x)$, and $\left\{\mathrm{succ}^N(n+2^{i-1})\right\}_{i=1,\ldots,m}$.

The object constitutes a Skip Graph. Membership vector of object $x$, $\mathrm{mv}(x)$, is the ID of $x$. Node $\mathrm{succ}^N(x)$ corresponding to the object $x$ holds a link to the node $\mathrm{succ}^N(y)$ of the adjacent object $y$ on the Skip Graph.

## 3 Evaluation

The overlay based on the proposed method and Chord to be compared are constructed and the average path length for key range queries is investigated by simulation experiment. First, assuming that the number of objects is $M = 10^3$, and the number of nodes $N = 10, 10^2, 10^3, 10^4$ and key range queries are performed. The range queries are performed $10^3$ times from randomly selected nodes. The key of the object is a random integer in the range $[0, 10^4)$. The range of the key $[x, y)$ is determined based on a random number, and its length is assumed to be $10^2$. At this time, for each of the proposed method and Chord, the average path length of the queries are examined. The experimental results are shown in the Fig. 1, 2.

Next, assuming that the number of nodes is $N = 10^3$, the number of objects $M = 10^2, 10^3, 10^4$ and key range queries are performed. At this time, for each of the proposed method and Chord, the average path length of the queries are examined. The experimental results are shown in the Fig. 3, 4.

According to Fig. 1–4, it can be confirmed that the average path length for key range queries is smaller in the proposed method than in Chord. In the case of performing range queries in Chord, it is necessary to iteratively search the key range, but the proposed method searches only the key at the beginning of the search range, until the key at the end of the range, and only have to broadcast the search query. For this reason, the proposed method seems to be able to suppress the average path length for range queries to be small compared to Chord.

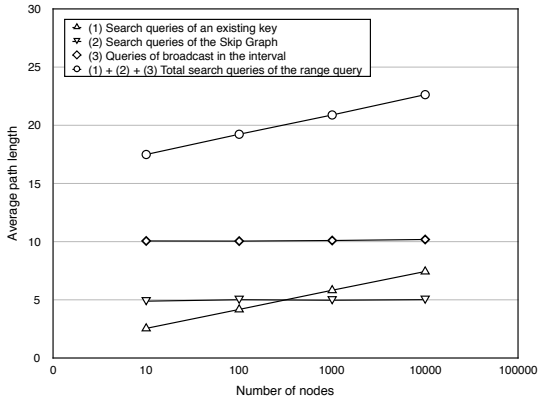


**Fig. 1.** Average path length of the range queries in the proposed method ($M = 10^3$).

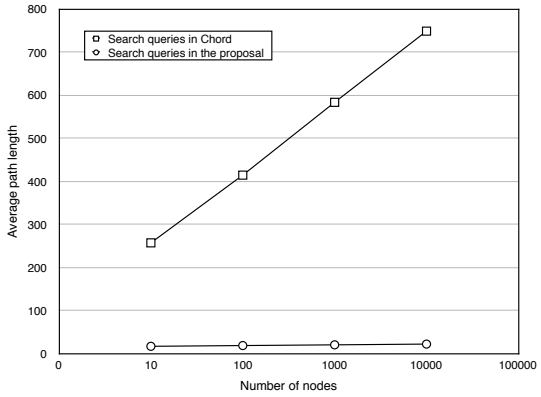


**Fig. 2.** Comparison of the average path length in the proposed method and Chord ($M = 10^3$).

## 4 Conclusion

We propose a method to distribute objects on each node in a range-queryable overlay. Simulation experiments shows that key range queries can be performed more efficiently than Chord. In the future, we evaluate the performance of the proposed method in case of the key distribution is not uniform and under churn.

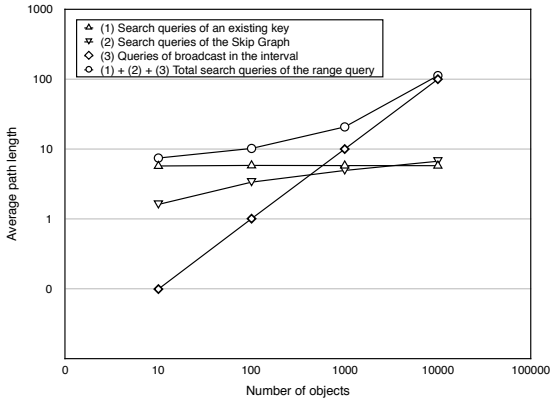


**Fig. 3.** Average path length of the range queries in the proposed method ($N = 10^3$).

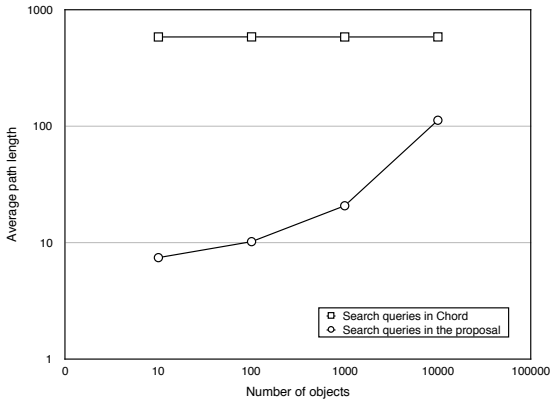


**Fig. 4.** Comparison of the average path length in the proposed method and Chord ($N = 10^3$).

## References

1. Ion Stoica, Robert Morris, David Karger, M Frans Kaashoek, Hari Balakrishnan, "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications," *SIGCOMM'01*, Vol. 31, No. 4, pp. 149–160, 2001.
2. David Karger, Eric Lehman, Tom Leighton, Rina Panigrahy, Matthew Levine, and Daniel Lewin, "Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web," *STOC '97*, pp. 654–663, 1997.
3. James Aspnes and Gauri Shah, "Skip Graphs," *SODA '03*, pp.384–393, 2003.
4. William Pugh, "Skip Lists: A Probabilistic Alternative to Balanced Trees," *Communications of the ACM*, Vol. 33, No. 6, pp. 668–676, 1990.